

BBBBBBBBBBBBBB		AAAAAAAAAA	CCCCCCCCCCCC	KKK	KKK	UUU	UUU	PPPPPPPPPPPP
BBBBBBBBBBBBBB		AAAAAAAAAA	CCCCCCCCCCCC	KKK	KKK	UUU	UUU	PPPPPPPPPPPP
BBBBBBBBBBBBBB		AAAAAAAAAA	CCCCCCCCCCCC	KKK	KKK	UUU	UUU	PPPPPPPPPPPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBBBBBBBBBBBBB		AAA	AAA	CCC	KKKKKKKKKK	UUU	UUU	PPPPPPPPPPPP
BBBBBBBBBBBBBB		AAA	AAA	CCC	KKKKKKKKKK	UUU	UUU	PPPPPPPPPPPP
BBBBBBBBBBBBBB		AAA	AAA	CCC	KKKKKKKKKK	UUU	UUU	PPPPPPPPPPPP
BBB	BBB	AAAAAAAAAAAAAAAA	CCC	KKK	KKK	UUU	UUU	PPP
BBB	BBB	AAAAAAAAAAAAAAAA	CCC	KKK	KKK	UUU	UUU	PPP
BBB	BBB	AAAAAAAAAAAAAAAA	CCC	KKK	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBBBBBBBBBBBBB		AAA	AAA	CCCCCCCCCCCC	KKK	UUUUUUUUUUUUUUUU	UUU	PPP
BBBBBBBBBBBBBB		AAA	AAA	CCCCCCCCCCCC	KKK	UUUUUUUUUUUUUUUU	UUU	PPP
BBBBBBBBBBBBBB		AAA	AAA	CCCCCCCCCCCC	KKK	UUUUUUUUUUUUUUUU	UUU	PPP

BBBBBBBB	AAAAAA	CCCCCCCC	KK	KK	DDDDDDDD	EEEEEEEEEE	FFFFFFFFFF	
BBBBBBBB	AAAAAA	CCCCCCCC	KK	KK	DDDDDDDD	EEEEEEEEEE	FFFFFFFFFF	
BB	BB	AA	AA	CC	DD	DD	EE	FF
BB	BB	AA	AA	CC	DD	DD	EE	FF
BB	BB	AA	AA	CC	DD	DD	EE	FF
BB	BB	AA	AA	CC	DD	DD	EE	FF
BBBBBBBB	AA	AA	CC	KKKKKK	DD	DD	EEEEEEEE	FFFFFFFF
BBBBBBBB	AA	AA	CC	KKKKKK	DD	DD	EEEEEEEE	FFFFFFFF
BB	BB	AAAAAAAAAA	CC	KK	DD	DD	EE	FF
BB	BB	AAAAAAAAAA	CC	KK	DD	DD	EE	FF
BB	BB	AA	CC	KK	DD	DD	EE	FF
BB	BB	AA	CC	KK	DD	DD	EE	FF
BBBBBBBB	AA	AA	CCCCCCCC	KK	DDDDDDDD	EEEEEEEEEE	FF	
BBBBBBBB	AA	AA	CCCCCCCC	KK	DDDDDDDD	EEEEEEEEEE	FF	

SSSSSSSS	DDDDDDDD	LL
SSSSSSSS	DDDDDDDD	LL
SS	DD	DD
SS	DD	DD
SS	DD	DD
SS	DD	DD
SSSSSS	DD	DD
SSSSSS	DD	DD
SS	DD	DD
SS	DD	DD
SS	DD	DD
SS	DD	DD
SSSSSSSS	DDDDDDDD	LLLLLLLLLL
SSSSSSSS	DDDDDDDD	LLLLLLLLLL


```
{ BACKDEF.SDL - BACKUP Media Structure Definitions
{
{ Version:      'V04-000'
{
{*****
{*
{* COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
{* DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
{* ALL RIGHTS RESERVED.
{*
{* THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
{* ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
{* INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
{* COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
{* OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
{* TRANSFERRED.
{*
{* THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
{* AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
{* CORPORATION.
{*
{* DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
{* SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
{*
{*****
{
{++
{ FACILITY: VMS BACKUP Utility
{
{ ABSTRACT:
{
{   This file contains the structure definitions for the BACKUP
{   media format.
{
{ ENVIRONMENT:
{
{   VAX/VMS operating system.
{
{ --
{
{ AUTHOR: Andrew C. Goldstein, CREATION DATE: 27-Aug-1980 15:40
{
{ MODIFIED BY:
{
{   V03-010 LY0510      Larry Yetto      19-JUL-1984 08:47
{   Increase size of DEVTYP field in physical volume attribute
{   record from 1 to 4
{
{   V03-009 LY0508      Larry Yetto      12-JUL-1984 10:13
{   Increase journal directory name length to be equal to MAXRSS.
{   MAXRSS was raised to 255 from 252 so we must follow.
{
{   V03-008 LY0505      Larry Yetto      11-JUL-1984 15:36
```


Correct spelling of BRH\$V_NONSEQUENTIAL

V03-007 ACG0433 Andrew C. Goldstein, 9-Jul-1984 18:37
Add per block error flags to BACKUP save set format;
general SDL cleanup

V03-006 LY0489 Larry Yetto 21-MAY-1984 16:11
Increase size of journal directory name entry to
allow long directory names.

V03-005 LY0462 Larry Yetto 1-FEB-1984 10:33
Add support for journal file structure level 2

V03-004 ACG0364 Andrew C. Goldstein, 10-Oct-1983 15:51
Add BACKUP format definitions for HSC backup

V03-003 JEP0003 J. Eric Pollack, 23-Apr-1983 10:35
Add support for encrypted savesets.

V03-002 ACG0332 Andrew C. Goldstein, 26-Apr-1983 19:15
Add highwater mark and journal flags file attributes

V03-001 LMP0044 L. Mark Pilant, 21-Oct-1982 14:00
Add support for saving and restoring ACL's.

V02-007 MLJ0081 Martin L. Jack, 26-Feb-1982 15:02
Add RETAINMIN and RETAINMAX attributes for new home block
fields.

V02-006 MLJ0075 Martin L. Jack, 28-Jan-1982 19:56
Add VERLIMIT and DIR_VERLIM attributes for version limit
handling.

V02-005 MLJ0062 Martin L. Jack, 3-Dec-1981 12:13
Add DIR_STATUS attribute to support /INCREMENTAL.

V02-004 MLJ0036 Martin L. Jack, 28-Aug-1981 17:14
Implement parent directory attributes.

V02-003 MLJ0023 Martin L. Jack, 23-Apr-1981 11:28
Implement placement attribute.

V02-002 MLJ0020 Martin L. Jack, 20-Apr-1981 21:42
Implement /JOURNAL qualifier.

V02-001 MLJ0010 Martin L. Jack, 25-Mar-1981 13:43
Add subfields to block header FID and DID. Add INDEXLBN,
BOOTVBN, BOOTBLOCK attributes for image restore. Add NUM_ATTRS
symbol. Change maximum length of BACKVER attribute.


```
{+
{
{ BBH - BACKUP block header. This structure prefixes each physical
{ record on the backup medium. It contains fields necessary for error
{ detection and recovery, and sufficient identification to allow the
{ block to be correctly interpreted in the absence of other information.
{
{-
```

```
module $BBHDEF;
```

```
aggregate BBHDEF structure fill prefix BBH$;
```

```
  SIZE word unsigned; /* size in bytes of block header
  OPSYS word unsigned; /* operating system ID
  SUBSYS word unsigned; /* subsystem ID
  APPLIC word unsigned; /* application ID
  NUMBER longword unsigned; /* block sequence number
  FILL_1 byte dimension 20 fill prefix BBHDEF tag $$; /* reserved
  constant "COMMON" equals . prefix BBH$ tag K; /* end of common header
  constant "COMMON" equals . prefix BBH$ tag C; /* end of common header

  STRUCLEV structure word unsigned; /* block structure level
  STRUCVER byte unsigned; /* structure version number
  STRUCLEV byte unsigned; /* major structure level
  constant LEVEL1 equals 257 prefix BBH tag $K; /* level 1, version 1
end STRUCLEV;
VOLNUM word unsigned; /* media volume number
CRC longword unsigned; /* block CRC
BLOCKSIZE longword unsigned; /* block size in bytes
FLAGS structure longword unsigned; /* block flags
  NOCRC bitfield mask; /* no CRC computed for block
end FLAGS;
SSNAME character length 32; /* save set name (counted ASCII)
/* *** Note: Old overlay form must be used in FID & DID due to SDL bug
/* *** that omits the field size symbol in a structured field.
  FID_OVERLAY union fill;
    FID word unsigned dimension 3; /* current file ID
    FID_FIELDS structure fill;
      FID_NUM word unsigned; /* file number
      FID_SEQ word unsigned; /* file sequence number
      FID_RVN_OVERLAY union fill;
        FID_RVN word unsigned; /* relative volume number
        FID_RVN_FIELDS structure fill;
          FID_RVN byte unsigned; /* alternate format RVN
          FID_NMX byte unsigned; /* alternate format file number extension
        end FID_RVN_FIELDS;
      end FID_RVN_OVERLAY;
    end FID_FIELDS;
  end FID_OVERLAY;
  DID_OVERLAY union fill;
    DID word unsigned dimension 3; /* current directory ID
    DID_FIELDS structure fill;
      DID_NUM word unsigned; /* file number
      DID_SEQ word unsigned; /* file sequence number
```



```

    DID_RVN_OVERLAY union fill;
        DID_RVN word unsigned;          /* relative volume number
        DID_RVN_FIELDS structure fill;
            DID_RVN byte unsigned;       /* alternate format RVN
            DID_NMX byte unsigned;       /* alternate format file number extension
        end DID_RVN_FIELDS;
    end DID_RVN_OVERLAY;
end DID_FIELDS;
end DID_OVERLAY;
FILENAME character length 128;          /* current file name
RTYPE byte unsigned;                   /* record type of current file
RATTRIB byte unsigned;                 /* record attributes of current file
RSIZE word unsigned;                   /* record size of current file
BKTSIZE byte unsigned;                 /* bucket size of current file
VFCSIZE byte unsigned;                 /* VFC area size of current file
MAXREC word unsigned;                  /* maximum record size of current file
FILESIZE longword unsigned;            /* allocation of current file
RESERVED2 character length 22;         /* reserved
CHECKSUM word unsigned;                 /* header checksum
constant "LENGTH" equals . prefix BBH$ tag K; /* length of block header
constant "LENGTH" equals . prefix BBH$ tag C; /* length of block header

end BBHDEF;
end_module $BBHDEF;
```



```
module $BRHDEF;
```

```
/**
/*
/* BRH - backup record header. This structure prefixes each record
/* within a data block. It identifies the type and use of the record.
/*
/*-
```

```
aggregate BRHDEF structure fill prefix BRH$;
```

```
  RSIZE word unsigned;
```

```
  RTYPE word unsigned;
```

```
  constant(
```

```
    NULL
```

```
    , SUMMARY
```

```
    , VOLUME
```

```
    , FILE
```

```
    , VBN
```

```
    , PHYSVOL
```

```
    , LBN
```

```
    , FID
```

```
    , FILE_EXT
```

```
    , LBN_576
```

```
  ) equals 0 increment 1 prefix BRH tag $K;
```

```
  FLAGS structure longword unsigned;
```

```
    BADDATA bitfield mask;
```

```
    DIRECTORY bitfield mask;
```

```
    NONSEQUENTIAL bitfield mask;
```

```
    BLOCKERRS bitfield mask;
```

```
  end FLAGS;
```

```
  "ADDRESS" longword unsigned;
```

```
  BLOCKFLAGS word unsigned;
```

```
  RESERVED word unsigned;
```

```
  constant "LENGTH" equals . prefix BRH$ tag K;
```

```
  constant "LENGTH" equals . prefix BRH$ tag C;
```

```
/* record size in bytes
```

```
/* record type
```

```
/* null record
```

```
/* BACKUP summary record
```

```
/* volume summary record
```

```
/* file attribute record
```

```
/* file virtual block record
```

```
/* physical volume attribute record
```

```
/* physical volume logical block record
```

```
/* file ID record
```

```
/* file attribute extension record
```

```
/* 576 byte LBN record
```

```
/* record flags
```

```
/* data error occurred in record
```

```
/* record belongs to a directory file
```

```
/* data record is not in LBN / VBN sequence
```

```
/* per block error mask present
```

```
/* address of data (e.g., VBN or LBN)
```

```
/* per block error mask
```

```
/* reserved
```

```
end BRHDEF;
```

```
end_module $BRHDEF;
```



```
module $BSADEF;
```

```
/**
```

```
/*
```

```
/* BACKUP attribute codes. These attributes appear in various records  
/* in a save set.
```

```
/*
```

```
/*-
```

```
/*
```

```
/* Definition for file ID record  
/* (Structure level definition shared with other attribute records)  
/*
```

```
aggregate BSADEF structure fill prefix BSA$;
```

```
    STRUCLEV word unsigned;
```

```
    FID_NUM word unsigned;
```

```
    FID_RVN structure word unsigned;
```

```
        FID_RVN byte unsigned;
```

```
        FID_NMX byte unsigned;
```

```
    end FID_RVN;
```

```
    FID_COUNT word unsigned;
```

```
    FID_SEQ word unsigned;
```

```
/* save set structure level
```

```
/* NUM word of file ID
```

```
/* RVN word of file ID
```

```
/* alternate format RVN word
```

```
/* alternate format RVN word
```

```
/* count of SEQ words following
```

```
/* SEQ word of file ID, repeated
```

```
/* BSA$W_FID_COUNT times
```

```
/*
```

```
/* Attribute entry header
```

```
/*
```

```
end BSADEF;
```

```
aggregate BSADEF1 structure fill prefix BSA$;
```

```
    SIZE word unsigned;
```

```
    TYPE word unsigned;
```

```
    constant "LENGTH" equals . prefix BSA$ tag K;
```

```
    constant "LENGTH" equals . prefix BSA$ tag C;
```

```
/* size of attribute value
```

```
/* attribute code
```

```
/* length of attribute descriptor
```

```
/* length of attribute descriptor
```

```
/*
```

```
/* Data item within placement attribute record
```

```
/*
```

```
end BSADEF1;
```

```
aggregate BSADEF2 structure fill prefix BSA$;
```

```
    PLC_PTR word unsigned;
```

```
    PLC_COUNT longword unsigned;
```

```
    PLC_LBN structure longword unsigned;
```

```
        PLC_LOLBN word unsigned;
```

```
        PLC_HILBN word unsigned;
```

```
    end PLC_LBN;
```

```
end BSADEF2;
```

```
/* placement pointer
```

```
/* count of placed blocks
```

```
/* LBN of placed blocks
```

```
/*
```



```

/*
/* Attribute type codes
/*

```

```

/*
/* BACKUP summary record
/*

```

```

constant(
    SSNAME          /* save set name
    , COMMAND       /* command line
    , COMMENT       /* user comment
    , USERNAME      /* name of user who wrote the set
    , USERUIC       /* UIC of user
    , DATE          /* date backup was done
    , OPSYS         /* operating system
    , SYSVER        /* operating system version
    , NODENAME      /* operating system node name
    , SIR           /* CPU system ID register
    , DRIVEID       /* ID of drive writing backup
    , BACKVER       /* version number of BACKUP
    , BLOCKSIZE     /* block size of save set
    , XORSIZE       /* size of each XOR group
    , BUFFERS       /* number of buffers
    , VOLSETNAM     /* volume set name
    , NVOLS         /* number of volumes in set
    , BACKSIZE      /* total file space in save set
    , BACKFILES     /* total number of files in save set

```

```

/*
/* Volume summary record
/*

```

```

    , VOLSTRUCT     /* volume structure level
    , VOLNAME       /* volume label
    , OWNERNAME     /* volume owner name
    , FORMAT        /* volume file format name
    , RVN           /* relative volume number
    , VOOWNER       /* volume owner UIC
    , PROTECT       /* volume protection mask
    , FILEPROT      /* volume default file protection
    , RECPROT       /* volume default record protection
    , VOLCHAR       /* volume characteristics bits
    , VOLDATE       /* volume creation date
    , WINDOW        /* default file window size
    , LRU LIM       /* default directory LRU limit
    , EXTEND        /* default file extend size
    , CLUSTER       /* storage map cluster factor
    , RESFILES      /* number of reserved files
    , VOLSIZE       /* original volume size in blocks
    , TOTSIZE       /* total file space in volume set
    , TOTFILES      /* total number of files in volume set
    , MAXFILES      /* maximum number of files allowed
    , MAXFILNUM     /* highest file number
    , SERIALNUM     /* pack serial number

```


/*
/* File attribute record
/*

, FILENAME	/* file name
, STRUCLEV	/* file structure level
, FID	/* file ID
, BACKLINK	/* directory ID back link
, FILESIZE	/* file size in blocks
, UIC	/* file owner UIC
, FPRO	/* file protection mask
, RPRO	/* record protection mask
, ACLEVEL	/* access level
, UCHAR	/* file characteristics
, RECATTR	/* record attributes area
, REVISION	/* revision number
, CREDATE	/* creation date
, REVDAT	/* revision date
, EXPDATE	/* expiration date
, BAKDATE	/* backup date

/*
/* Physical volume attribute record
/*

, SECTORS	/* sectors per track
, TRACKS	/* tracks per cylinder
, CYLINDERS	/* cylinders per volume
, MAXBLOCK	/* number of logical blocks per volume
, DEVTYP	/* device type
, SERIAL	/* serial number
, DEVNAM	/* device name
, LABEL	/* label
, BADBLOCK	/* bad block descriptor, a sequence of
	/* pairs of longwords where the first
	/* is an LBN, the second is a count

/*
/* Additions
/*

, INDEXLBN	/* (VS) Index file bitmap starting LBN
, BOOTBLOCK	/* (VS) Boot block image
, BOOTVBN	/* (FA) VBN within file for boot block
, PLACEMENT	/* (FA) Placement data
, DIR_UIC	/* (FA) UIC of directory
, DIR_FPRO	/* (FA) Protection of directory
, DIR_STATUS	/* (FA) Status of directory
, DIR_VERLIM	/* (FA) Version limit of directory
, VERLIMIT	/* (FA) File version limit
, RETAINMIN	/* (VS) Minimum file retention period
, RETAINMAX	/* (VS) Maximum file retention period
, ACLSEGMENT	/* (FA) ACL segment for the file
, HIGHWATER	/* (FA) Highwater mark
, JNL_FLAGS	/* (FA) Journal control flags


```
      , CRYPDATKEY
      , LBNSIZE          /* (BS) File encryption attribute
                        /* (PS) Disk block size in bytes
      , NUM_ATTRS        /* number of attribute codes
      ) equals 1 increment 1 prefix BSA tag $K;

/*
/* Placement data type codes
/*

constant(
  , PLC_FID              /* file ID
  , PLC_COUNT            /* count of unplaced blocks
  , PLC_PLACE            /* placement pointer, count of placed blocks
  , PLC_PLLBN            /* placement pointer, count, LBN of placed blocks
  ) equals 1 increment 1 prefix BSA tag $K;
```



```
/*
/* Lengths of above attributes
/*
```

```
/*
/* BACKUP summary record
/*
```

```
constant SSNAME      equals 32 prefix BSA tag $$; /* save set name
constant COMMAND     equals 512 prefix BSA tag $$; /* command line
constant COMMENT     equals 512 prefix BSA tag $$; /* user comment
constant USERNAME    equals 32 prefix BSA tag $$; /* name of user who wrote the set
constant USERUIC     equals 4  prefix BSA tag $$; /* UIC of user
constant DATE        equals 8  prefix BSA tag $$; /* date backup was done
constant OPSYS       equals 2  prefix BSA tag $$; /* operating system
constant SYSVER      equals 4  prefix BSA tag $$; /* operating system version
constant NODENAME    equals 12 prefix BSA tag $$; /* operating system node name
constant SIR         equals 4  prefix BSA tag $$; /* CPU system ID register
constant DRIVEID     equals 16 prefix BSA tag $$; /* ID of drive writing backup
constant BACKVER     equals 32 prefix BSA tag $$; /* version number of BACKUP
constant BLOCKSIZE   equals 4  prefix BSA tag $$; /* block size of save set
constant XORSIZE     equals 2  prefix BSA tag $$; /* size of each XOR group
constant BUFFERS     equals 2  prefix BSA tag $$; /* number of buffers
constant VOLSETNAM    equals 12 prefix BSA tag $$; /* volume set name
constant NVOLS       equals 2  prefix BSA tag $$; /* number of volumes in set
constant BACKSIZE    equals 8  prefix BSA tag $$; /* total file space in save set
constant BACKFILES   equals 4  prefix BSA tag $$; /* total number of files in save set
```

```
/*
/* Volume summary record
/*
```

```
constant VOLSTRUCT   equals 2  prefix BSA tag $$; /* volume structure level
constant VOLNAME      equals 12 prefix BSA tag $$; /* volume label
constant OWNERNAME    equals 12 prefix BSA tag $$; /* volume owner name
constant FORMAT       equals 12 prefix BSA tag $$; /* volume file format name
constant RVN          equals 2  prefix BSA tag $$; /* relative volume number
constant VOOWNER      equals 4  prefix BSA tag $$; /* volume owner UIC
constant PROTECT      equals 2  prefix BSA tag $$; /* volume protection mask
constant FILEPROT     equals 2  prefix BSA tag $$; /* volume default file protection
constant RECPROT      equals 2  prefix BSA tag $$; /* volume default record protection
constant VOLCHAR      equals 2  prefix BSA tag $$; /* volume characteristics bits
constant VOLDATE      equals 8  prefix BSA tag $$; /* volume creation date
constant WINDOW       equals 1  prefix BSA tag $$; /* default file window size
constant LRU LIM      equals 1  prefix BSA tag $$; /* default directory LRU limit
constant EXTEND       equals 2  prefix BSA tag $$; /* default file extend size
constant CLUSTER      equals 2  prefix BSA tag $$; /* storage map cluster factor
constant RESFILES     equals 2  prefix BSA tag $$; /* number of reserved files
constant VOLSIZ       equals 4  prefix BSA tag $$; /* original volume size in blocks
constant TOTSIZE      equals 8  prefix BSA tag $$; /* total file space in volume
constant TOTFILES     equals 4  prefix BSA tag $$; /* total number of files in volume
constant MAXFILES     equals 4  prefix BSA tag $$; /* maximum number of files allowed
constant MAXFILNUM    equals 4  prefix BSA tag $$; /* highest file number
constant SERIALNUM    equals 4  prefix BSA tag $$; /* pack serial number
```



```
/*
/* File attribute record
/*
```

```
constant FILENAME equals 128 prefix BSA tag $$/* file name
constant STRUCLEV equals 2 prefix BSA tag $$/* file structure level
constant FID equals 6 prefix BSA tag $$/* file ID
constant BACKLINK equals 6 prefix BSA tag $$/* directory ID back link
constant FILESIZE equals 4 prefix BSA tag $$/* file size in blocks
constant UIC equals 4 prefix BSA tag $$/* file owner UIC
constant FPRO equals 2 prefix BSA tag $$/* file protection mask
constant RPRO equals 2 prefix BSA tag $$/* record protection mask
constant ACLEVEL equals 1 prefix BSA tag $$/* access level
constant UCHAR equals 4 prefix BSA tag $$/* file characteristics
constant RECATTR equals 32 prefix BSA tag $$/* record attributes area
constant REVISION equals 2 prefix BSA tag $$/* revision number
constant CREDATE equals 8 prefix BSA tag $$/* creation date
constant REVDATE equals 8 prefix BSA tag $$/* revision date
constant EXPDATE equals 8 prefix BSA tag $$/* expiration date
constant BAKDATE equals 8 prefix BSA tag $$/* backup date
```

```
/*
/* Physical volume attribute record
/*
```

```
constant SECTORS equals 1 prefix BSA tag $$/* sectors per track
constant TRACKS equals 1 prefix BSA tag $$/* tracks per cylinder
constant CYLINDERS equals 2 prefix BSA tag $$/* cylinders per volume
constant MAXBLOCK equals 4 prefix BSA tag $$/* number of logical blocks per volume
constant DEVTYP equals 4 prefix BSA tag $$/* device type
constant SERIAL equals 4 prefix BSA tag $$/* serial number
constant DEVNAM equals 64 prefix BSA tag $$/* device name
constant LABEL equals 12 prefix BSA tag $$/* label
constant BADBLOCK equals 8 prefix BSA tag $$/* bad block descriptor (one pair)
```

```
/*
/* Additions
/*
```

```
constant INDEXLBN equals 4 prefix BSA tag $$/* (VS) Index file bitmap starting LBN
constant BOOTBLOCK equals 512 prefix BSA tag $$/* (VS) Boot block image
constant BOOTVBN equals 4 prefix BSA tag $$/* (FA) VBN within file for boot block
constant PLACEMENT equals 2048 prefix BSA tag $$/* (FA) Placement data
constant DIR_UIC equals 4 prefix BSA tag $$/* (FA) UIC of directory
constant DIR_FPRO equals 2 prefix BSA tag $$/* (FA) Protection of directory
constant DIR_STATUS equals 1 prefix BSA tag $$/* (FA) Status of directory
constant DIR_VERLIM equals 2 prefix BSA tag $$/* (FA) Version limit of directory
constant VERLIMIT equals 2 prefix BSA tag $$/* (FA) File version limit
constant RETAINMIN equals 8 prefix BSA tag $$/* (VS) Minimum file retention period
constant RETAINMAX equals 8 prefix BSA tag $$/* (VS) Maximum file retention period
constant ACLSEGMENT equals 380 prefix BSA tag $$/* (FA) ACL segment for the file
constant HIGHWATER equals 4 prefix BSA tag $$/* (FA) Highwater mark
constant JNL_FLAGS equals 2 prefix BSA tag $$/* (FA) Journal control flags
constant CRYPDATKEY equals 24 prefix BSA tag $$/* (BS) Saveset encryption key
constant LBNSIZE equals 2 prefix BSA tag $$/* (PS) Disk block size in bytes
```



```
/*
/* Placement data type codes
/*

    constant PLC_FID      equals 6  prefix BSA tag $$; /* file ID
    constant PLC_COUNT    equals 4  prefix BSA tag $$; /* count of unplaced blocks
    constant PLC_PLACE    equals 6  prefix BSA tag $$; /* placement pointer, count of placed blocks
    constant PLC_PLLBN    equals 10 prefix BSA tag $$; /* placement pointer, count, LBN of placed blocks

{
{ Fields of a CRYPDATKEY attribute record
{
aggregate BSADEF3 structure fill prefix BSAS;
    CRYPTYP byte unsigned; /* Byte code describing saveset encr alg
    constant(
        CRYPT_DESCBC /* Code for DES Cypher Block Chaining
        ,CRYPT_DESECB /* Code for DES Electronic Code Book.
    ) equals 1 increment 1 prefix BSAS tag K;
    CRYPTPRES0 character length 3 fill; /* Unused
    CRYPTKEY quadword unsigned; /* DES key used to encrypt saveset
    CRYPTIV quadword unsigned; /* DES initialize vector prototype
    CRYPTCKSM longword unsigned; /* Checksum for attribute record
end BSADEF3;

end_module $BSADEF;
```



```
module $BJLDEF;
```

```
/*  
/*  
/* BACKUP journal file.  
/*  
/*-
```

```
aggregate BJLDEF structure fill prefix BJL$;
```

```
  SIZE byte unsigned;  
  TYPE byte unsigned;
```

```
/* record length (XOR format)  
/* record type
```

```
  constant(  
    STRUCLEV
```

```
    , SSNAME  
    , VOLUME  
    , DIRECTORY  
    , FILE
```

```
/* structure level  
/* save set name  
/* volume identification  
/* directory name  
/* file name
```

```
  ) equals 0 increment 1 prefix BJL tag $K;
```

```
  DATA byte unsigned;
```

```
/* beginning of data portion of record
```

```
/*  
/* Following definitions are all relative to data portion of record.  
/*  
/* Structure level record.  
/*
```

```
end BJLDEF;
```

```
aggregate BJLDEF1 structure fill prefix BJL$;
```

```
  STRUCLEV structure word unsigned;
```

```
  STRUCVER byte unsigned;
```

```
  STRUCLEV byte unsigned;
```

```
  constant LEVEL1 equals 257 prefix BJL tag $K;
```

```
  constant LEVEL2 equals 258 prefix BJL tag $K;
```

```
/* structure level  
/* structure version number  
/* major structure level  
/* level 1, version 1  
/* level 1, version 2
```

```
end STRUCLEV;
```

```
  constant STRUC_LEN equals . prefix BJL$ tag K;
```

```
  constant STRUC_LEN equals . prefix BJL$ tag C;
```

```
/* length of record  
/* length of record
```

```
/*  
/* Save set name record.  
/*
```

```
end BJLDEF1;
```

```
aggregate BJLDEF2 structure fill prefix BJL$;
```

```
  CREFDATE quadword unsigned;
```

```
  SSNAME character length 32;
```

```
  constant SSNAME_LEN equals . prefix BJL$ tag K;
```

```
  constant SSNAME_LEN equals . prefix BJL$ tag C;
```

```
/* creation date and time  
/* save set name (variable length)  
/* maximum length of record  
/* maximum length of record
```

```
/*  
/* Volume identification record.  
/*
```


end BJJDEF2;

aggregate BJJDEF3 structure fill prefix BJJ\$;

VOLNAME character length 12;

/* volume label

VOLNUMBER word unsigned;

/* volume sequence number

constant VOLUME_LEN equals . prefix BJJ\$ tag K;

/* length of record

constant VOLUME_LEN equals . prefix BJJ\$ tag C;

/* length of record

/*

/* Directory record.

/*

end BJJDEF3;

aggregate BJJDEF4 structure fill prefix BJJ\$;

DIRNAME character length 255;

/* directory name (variable length)

constant DIR_LEN equals . prefix BJJ\$ tag K;

/* maximum length of record

constant DIR_LEN equals . prefix BJJ\$ tag C;

/* maximum length of record

/* stored in XOR format

/*

/* File record.

/*

end BJJDEF4;

/* The following masks define flag bits. The structure is being defined

/* in this way because the flags do not have a set position in the record

/* defined by BJJDEF5 but simply follow directly after the file name.

/* By defining the structures in this manner there is a full longword

/* reserved but is not position dependant

aggregate FLAG_BITS structure fill prefix BJJ\$;

HEADONLY bitfield mask ;

/* NOBACKUP was set when file was saved

end FLAG_BITS ;

aggregate BJJDEF5 structure fill prefix BJJ\$;

FILENAME character length 128;

/* file name (variable length)

FLAGS longword unsigned fill;

/* flags

constant FILE_LEN equals . prefix BJJ\$ tag K;

/* maximum length of record

constant FILE_LEN equals . prefix BJJ\$ tag C;

/* maximum length of record

end BJJDEF5;

end_module \$BJJDEF;

0009 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY